Kotlin - Keywords

Kotlin keywords are predefined, reserved words used in Kotlin programming that have special meanings to the compiler. These words cannot be used as an identifier (variables names, package names, function names etc.) and if used then compiler will raise an exception.

Kotlin uses **fun** keyword to define a function, so if we we will try to use it as a variable name then it will be an exception. For example:

```
fun main() {
   var fun = "Zara Ali"  // Not allowed, throws an exception
   var age = 19          // Valid variable name

   println("Name = $fun")
   println("Age = $age")
}
```

When you run the above Kotlin program, it will generate the following output:

```
main.kt:2:7: error: expecting property name or receiver type
var fun = "Zara Ali" // Not allowed, throws an exception
^
main.kt:2:11: error: expecting '('
var fun = "Zara Ali" // Not allowed, throws an exception
^
main.kt:5:21: error: keyword cannot be used as a reference
println("Name = $fun")
^
```

Kotlin keywords have been categorised into three broad categories: (a) Hard Keywords (b) Soft Keywords (c) Modifier Keywords

As a good programming practice, it is highly recommended not to use any of the mentioned keywords to name any identifiers while coding in Kotlin.

(a) Kotlin Hard Keywords

Following is a list of hard keywords and they cannot be used as identifiers:

as	as?	break	class
continue	do	else	false
for	fun	if	in
!in	interface	is	!is
null	object	package	return
super	this	throw	true
try	typealias	typeof	val
var	when	while	

(b) Kotlin Soft Keywords

Following is the list of keywords (soft) in the context when they are applicable and can be used as identifiers in other contexts:

by	catch	constructor	delegate
dynamic	field	file	finally
get	import	init	param
property	receiver	set	setparam
value	where		

(c) Kotlin Modifier Keywords

Following is the list of tokens which act as keywords in modifier lists of declarations and can be used as identifiers in other contexts:

actual	abstract	annotation	companion
const	crossinline	data	enum
expect	external	final	infix
inline	inner	internal	lateinit
noinline	open	operator	out
override	private	protected	public
reified	sealed	suspend	tailrec
vararg			

Quiz Time (Interview & Exams Preparation)

Q 1 - Which of the following is a hard keyword in Kotlin:

A - var

B - val

C - for

D - All of the above

Q 2 - Identify which line of the following program will raise an error:

```
var name = "Zara Ali"
var age = 19
var class = "6th"
var height = 5.3
```

A - First Line

B - Second Line

C - Third Line

D - Last Line

Q 3 - Which statment is incorrect in Kotlin

A - We can use Kotlin hard keywords to define any identifier

B - Kotlin allows using soft keywords as identifiers based on the context.

C - It is a good practice not to use any type of keywords while naming an identifier.

D - All of the above statements are incorrect